UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
12/576,140	10/08/2009	Kumar Rajamani	50277-3709	1462

42425 7590 12/02/2016
HICKMAN PALERMO BECKER BINGHAM/ORACLE
1 Almaden Boulevard
Floor 12
SAN JOSE, CA 95113

EXAMINER	
PARK, GRACE A	

ART UNIT	PAPER NUMBER
2157	

NOTIFICATION DATE	DELIVERY MODE
12/02/2016	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

usdocket@h35g.com

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Ex parte KUMAR RAJAMANI, HOCHAK HUNG,
JAEBOCK LEE, and PHILIP YAM

Appeal 2015-007298
Application No. 12/576,140[1]
Technology Center 2100

Before ST. JOHN COURTENAY, III, MARC S. HOFF, and
SCOTT B. HOWARD, *Administrative Patent Judges*.

HOFF, *Administrative Patent Judge*.

DECISION ON APPEAL

STATEMENT OF THE CASE

Appellants appeal under 35 U.S.C. § 134 from a Final Rejection of claims 1, 12–34, 37, and 38.[2] We have jurisdiction under 35 U.S.C. § 6(b).

We affirm-in-part.

Appellants' invention is a method for managing objects in a volatile memory cache. A mapping is created between a base object and a mapped object that persists while the objects reside in the cache. Spec. ¶ 18. The

[1] The real party in interest is Oracle International Corporation.
[2] Claims 2–11, 35, and 36 been cancelled.

type of mapping defines how the objects are treated when the mapped object is created, read, or written. At read and write time, the mapping type may define on which object a mutual exclusion lock is held, which content is returned, or which object's content is updated. *See* Abstract.

Claim 1 is exemplary of the claims on appeal:

> 1. A computer-implemented method for operating on stored objects, the method comprising:
>
> storing a base object handle representing a base object, the base object handle comprising a base object content association that links the base object to a base object content;
>
> storing a mapped object handle representing a mapped object,
>
> the mapped object handle comprising:
>
> a mapping that identifies the base object handle, and
>
> a mutable mapping type selected from a plurality of mapping types, each of the plurality of mapping types defining semantics of one or more operations on the mapped object;
>
> receiving a request to perform an operation on the mapped object;
>
> responding to the request to perform the operation by:
>
> determining one or more actions to perform based on the semantics of the operation defined by the mapping type of the mapped object handle;
>
> and
>
> performing the one or more actions;
>
> wherein the method is performed by one or more computing devices.

The Examiner relies upon the following prior art in rejecting the claims on appeal:

Claims 1, 15–17, 24, 26, 29, and 30 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Bernstein (US 2004/0162841 Al; pub. Aug. 19, 2004) and Kaakani (US 2007/0226685 Al; pub. Sept. 27, 2007).

Claim 12 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over Bernstein, Kaakani, Chiu (US 6,654,029 B1; iss. Nov. 25, 2003), and Ozbutun (US 2005/0240570 Al; pub. Oct. 27, 2005).

Claim 13 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over Bernstein, Kaakani, and Attaluri (US 5,897,634; iss. Apr. 27, 1999).

Claims 14 and 28 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Bernstein, Kaakani, Chiu, and Attaluri.

Claims 18–23 and 31–34 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Bernstein, Kaakani, and Ozbutun.

Claims 25 and 27 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Bernstein, Kaakani, and Chiu.

Claims 37 and 38 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Bernstein, Kaakani, and Naga (US 2011/0191544 Al; pub. Aug. 4, 2011).

Throughout this decision, we make reference to the Appeal Brief (“App. Br.,” filed Feb. 24, 2015), the Reply Brief (“Reply Br.,” filed Aug. 3, 2015), and the Examiner’s Answer (“Ans.,” mailed June 2, 2015) for their respective details.

ISSUES

1. Does the combination of Bernstein and Kaakani teach or suggest storing a base object handle representing a base object, the mapped object handle comprising a mapping that identifies the base object handle?

2. Does the combination of Bernstein and Kaakani teach or suggest a mutable mapping type selected from a plurality of mapping types, each of the plurality of mapping types defining semantics of one or more operations on the mapped object?

3. Does the combination of Bernstein and Kaakani teach or suggest determining one or more actions to perform based on the semantics of the operation defined by the mapping type of the mapped object handle?

4. Does the combination of Bernstein, Kaakani, and Ozbutun teach or suggest a syntactic or semantic mapping type, wherein the base object content includes an evaluation of a function applied to a mapped object handle name?

5. Does the combination of Bernstein, Kaakani, and Chiu teach or suggest a plurality of mapping types including a versionable type and a non-versionable type?

PRINCIPLES OF LAW

Our reviewing court has held that "[a] reference may be said to teach away when a person of ordinary skill, upon reading the reference, would be discouraged from following the path set out in the reference, or would be led in a direction divergent from the path that was taken by the applicant." *In re Gurley*, 27 F.3d 551, 553 (Fed. Cir. 1994); *Para-Ordnance Mfg. v. SGS Importers Int'l,* 73 F.3d 1085, 1090 (Fed. Cir. 1995).

ANALYSIS

REJECTIONS OVER BERNSTEIN AND KAAKANI

CLAIMS 1 AND 29

We are not persuaded by Appellants' argument that Bernstein fails to teach a mapped object handle. *See* App. Br. 12. Appellants define a handle as "an opaque, indirect reference to a region of memory that is managed by the object system. A handle may be used by a program that is not part of the object system to specify operations to be performed on the object. When used in the context of the object system, the word handle refers to the memory that is referenced by the handle. When discussed from the perspective of the program using the object, handle refers to the object reference." Spec. ¶ 20. We agree with the Examiner that "a person of ordinary skill in the art would understand that structures are referenced using stored handles (e.g., a type and identifier)" (Final Act. 2), and that "a specific data structure would be referenced using a specific handle" (Final Act. 3).

We further agree with the Examiner that because the claims do not specifically indicate the context in which the claimed limitations are performed, a handle can be interpreted as referring either to (a) a data structure or (b) a reference to a data structure. *See* Spec. ¶ 20; Ans. 3.

As noted by the Examiner, Bernstein discloses that "[a]n object, as is known in the art, is a data structure that has a persistent state. The persistent state consists of attributes, which comprise scalar values and object references. A scalar value is a value such as a string, integer or Boolean. An object reference specifies one side of a binary relationship between two objects that refer to each other. In other words, the reference is to another object, which in turn refers back to the referring object. Each attribute is identified by a name, and each attribute has a data type. The data type for an attribute identifies either the type of scalar value for the attribute or the type of relationship defined by the attribute." Ans. 3, *citing* Bernstein ¶ 4.

The data structure of Bernstein may not be called a "handle" within that reference, but said data structure corresponds to the claimed handle. It represents an object, and performs the function of a handle, by storing metadata such as a reference ("mapping" or pointer ¶ 44) to another object.

With respect to the claim term "semantics," the Examiner finds, and Appellants do not traverse, that semantics refers to the study of meaning, and thus that "semantics of one or more operations on the mapped object" may be interpreted as something that indicates which operations can be performed on the mapped object. Ans. 6. As known in the art, we agree with the Examiner that Bernstein teaches that "[a]n object is typically an instance of a class. A class is a body of code that implements one or more object types," and "[t]he types of operations performed vary depending on the class." Ans. 6–7; Bernstein ¶ 9. We thus agree with the Examiner's further finding that an object type is implemented by a class that includes code to perform various operations on an object. Ans. 6.

With respect to the claimed "mutable mapping type," we also agree with the Examiner that Kaakani teaches that "the class (i.e., mapping type) of an object can be changed" – that is, that it is mutable. Ans. 6; Kaakani ¶ 32. Appellants argue that this teaching does not correspond to the claimed mutable mapping type of the mapped object handle, but provide no evidence in support of their position. App. Br. 14; Reply Br. 5.

We do not agree with Appellants' argument that Bernstein does not teach "determining one or more actions to perform based on the semantics of the operation defined by the mapping type of the mapped object handle." App. Br. 14. Appellants again argue that "[n]o semantics of any operation are defined by any mapping type in Bernstein" and "[n]o mapping type is described as part of an object handle representing an object in Bernstein." *Id.* However, we again agree with the Examiner that Bernstein teaches that a mapping type "is implemented by a class that includes code to perform various operations," again corresponding to "semantics of one or more operations," on a mapped object, as discussed *supra*. *See* Ans. 7, citing Bernstein ¶ 9.

Thus, we find that the Examiner did not err in combining Bernstein and Kaakani to obtain the claimed invention. We sustain the Examiner's § 103 rejection of claims 1 and 29.

CLAIMS 15–17

Appellants do not present separate argument for the patentability of claims 15–17, relying on the arguments made with respect to claim 1's recitation of "defining semantics of one or more operations." App. Br. 15–17. Because we find *supra* that Bernstein teaches this limitation, we sustain the Examiner's § 103 rejection of claims 15–17, for the same reasons given with respect to the Examiner's rejection of parent claim 1.

CLAIM 24

We are not persuaded by Appellants' argument that "the simple manipulation of cached objects by applications is not adequate disclosure or suggestion to render obvious specifically altering the mapping type of the object in the cache." App. Br. 17. We agree with the Examiner that Appellants' argument is directed to Bernstein alone, rather than the combined teachings of Bernstein and Kaakani. As noted *supra*, we agree with the Examiner that Kaakani teaches that the class (i.e., mapping type) of an object can be changed. *See* Kaakani ¶ 32. The combined teachings of Bernstein (that an object can be cached) and Kaakani (that the mapping type can be changed) therefore suggest the claimed invention. *See* Ans. 8. We sustain the Examiner's § 103 rejection of claim 24.

CLAIM 26

We are not persuaded by Appellants' argument that while Bernstein teaches that objects may be stored in a combination of caches., such is not the same as "a mapped object that is mapped to a base object resides in a first cache, and the base object resides in a second cache." App. Br. 18.

We agree with the Examiner that Bernstein teaches that "[t]he prefetched objects or object data are stored in caches for later access." In one embodiment, "a client cache 220, a server cache 255, and a storage cache 275 are used to store prefetched object data." Bernstein ¶ 42; *see* Ans. 9. The Examiner then further finds, and we agree, that the person of ordinary skill would recognize that a mapped object and a base object could be stored in the same cache, or different caches, subject to a limited number of combinations. *See* Ans. 9.

Therefore, we agree with the Examiner's conclusion that it would have been obvious to select one of the limited number of conclusions, i.e., storing a mapped object in one cache and a base object in a second cache, as recited in claim 26. We sustain the Examiner's § 103 rejection over Bernstein and Kaakani.

CLAIMS 12 AND 30

Appellants do not present separate argument for the patentability of claims 12 and 30. Appellants merely repeat the argument made with respect to claim 1 that none of the references discloses a mapping type, selected from a plurality of mapping types, each of the plurality of mapping types defining semantics of one or more operations on the mapped object. App. Br. 18–19; *see* Ans. 9. Thus, because we find *supra* that Bernstein teaches this limitation, we sustain the Examiner's § 103 rejection of claim 12 over Bernstein and Kaakani, and the Examiner's § 103 rejection of claim 30 over Bernstein, Kaakani, Chiu, and Ozbutun, for the same reasons given *supra* with respect to the Examiner's rejection of parent claim 1 (over Bernstein and Kaakani).

CLAIMS 13, 14, AND 28

Appellants do not present separate argument for the patentability of claims 13, 14, and 28, relying instead on the arguments made with respect to claim 1 regarding the phrase "defining semantics of one or more operations." App. Br. 19–22; *see* Ans. 9–10. Appellants merely make mention of what each claim recites, and provide unsupported allegations that the combination of references fails to teach what is claimed. *Id.* Because we find *supra* that Bernstein teaches this limitation, we sustain the Examiner's § 103 rejection of claim 13 over Bernstein, Kaakani, and Attaluri, and we sustain the Examiner's § 103 rejection of claims 14 and 28 over Bernstein, Kaakani, Chiu, and Attaluri, for the same reasons given *supra* with respect to the Examiner's rejection of independent claim 1 over Bernstein and Kaakani.

REJECTIONS OVER BERNSTEIN, KAAKANI, AND OZBUTUN

CLAIMS 18–20 AND 31–33

Claims 18 and 31 recite "wherein the mapping type is a syntactic relationship type, the mapped object handle includes a name, and the base object content includes an evaluation of a function applied to the name."

The Examiner finds that Ozbutun teaches that cached results sets may be associated with each other in various relationships. Final Act. 15, citing Ozbutun ¶¶ 31–33 and 49. Appellants' Specification defines "syntactic relationship" as "one in which the name of the mapped object may be rewritten by a set of rules to match the content of a base object." Spec. ¶ 33. We have reviewed paragraphs 31–33 and 49 of Ozbutun, and we do not find a teaching that the name of a mapped object may be rewritten by a set of rules to match the content of a base object.

Accordingly, we do not sustain the Examiner's § 103 rejection of claims 18 and 31, nor the rejection of claims 19, 20, 32, and 33 dependent therefrom.

CLAIMS 21–23 AND 34

Claims 21 and 34 recite "wherein the mapping type is a semantic relationship type, the mapped object handle includes a name, and the base object content includes an evaluation of a function applied to the name."

The Examiner finds that Ozbutun teaches that cached results sets may be associated with each other in various relationships. Final Act. 16, citing Ozbutun ¶¶ 31–33 and 49. Appellants' Specification defines "semantic relationship" as "similar to a syntactic relationship" but "requires only that the results of the evaluation be the same, not that the expression be syntactically equivalent. That is, simple rewriting rules like removing white space or ignoring differences in capitalization cannot reconcile the differences between two expressions presented by cached objects. However, the expressions of the mapped and base objects must have the same meaning such that the evaluation of each yields the same results." Spec. ¶ 36.

While the pertinent sections of Ozbutun do teach that "the results set of the similar operation may be used instead of executing the actual operation," in response to "queries [that] are similar but not identical," (Ozbutun ¶ 49), we do not find in Ozbutun a teaching of a semantic relationship that is similar to a syntactic relationship, involving the rewriting of the name of a mapped object to match the content of a base object. Accordingly, we do not sustain the Examiner's § 103 rejection of claims 21 and 34, nor the rejection of claims 22 and 23 dependent from claim 21.

REJECTIONS OVER BERNSTEIN, KAAKANI, AND CHIU

CLAIM 27

Appellants present several arguments for the patentability of claim 27. Several of these arguments – regarding storing a first object handle representing a base object; a second object handle comprising a mapping that identifies the first object handle; a mutable mapping type selected from a plurality of mapping types, each of the plurality defining semantics of one or more operations on the mapped object – are a repetition of arguments made with respect to claim 1. *See* App. Br. 24–26. These arguments are not persuasive for reasons expressed *supra* in the analysis of claim 1.

Appellants further assert that Bernstein, Kaakani, and Chiu fail to teach or suggest a plurality of mapping types including "a versionable type and a non-versionable type." Appellants contend that Chiu teaches away from "a mutable mapping type selected from a plurality of mapping types," because Chiu teaches that assets are declared as either versioned or versioned at creation time. *See* App. Br. 26–27.

According to Appellants' Specification, a "non-versionable relationship" between cached objects "is one in which a mapped object shares the contents of a base object such that writing the mapped object updates the contents of the base object." Spec. ¶ 29. A "versionable relationship provides for copy-on-write semantics." Spec. ¶ 30. In Appellants' example, a versionable mapping is created from Object A to Object B. "When a write request is received for updating A, an exclusive write lock is obtained on A, but a shared read lock is obtained on Object B. Thus, other objects with versionable mappings to Object B may continue to access Object B's content while Object B's content is copied into Object A . . . other mappings to Object B are not blocked or disturbed by writing through a versionable mapping to Object B." *Id.*

Appellants note the Examiner's reliance on Chiu to teach that "[a]n asset is typically declared as either Versioned or unversioned at creation time and it stays Versioned or unversioned from that point on." App. Br. 26, citing Chiu 16:18–29. Appellants argue, without further evidence or explanation, that Chiu thus teaches away from a mutable mapping type. App. Br. 26. We disagree with Appellants, because Chiu only states that assets are *typically* declared either versioned or unversioned. We find that Chiu thus does not discourage one of ordinary skill in the art from declaring assets versioned or unversioned such that the asset type is changeable (i.e., mutable).

Appellants' arguments concerning claim 25 merely refer to the arguments made with respect to claims 1 and 27. App. Br. 27–28.

Because we do not agree with Appellants that the Examiner erred, we sustain the Examiner's § 103 rejection of claims 25 and 27 over Bernstein, Kaakani, and Chiu.

REJECTION OVER BERNSTEIN, KAAKANI, AND NAGA
CLAIMS 37 AND 38

Appellants' argument that "removal of objects from a cache is not equivalent to retaining a mapping relationship between a mapped object and a base object that is stored in a mapped object handle" is not persuasive to show that the Examiner erred. *See* App. Br. 29. We agree with the Examiner that an argument against Naga alone does not establish error in a rejection based upon a combination of references. We agree with the Examiner that Bernstein teaches storing mapping information of an object with the object, and that objects may be stored in a combination of caches; and that Naga discloses removing related objects from a cache. Ans. 11; Naga ¶¶ 31–33; Bernstein ¶¶ 3–5, 42. We agree that in accordance with Naga's teaching, related (i.e. mapped) objects retain mapping information while they are stored in the cache. *See* Ans. 11.

With respect to claim 38, as with claim 26 *supra*, we conclude that given the presence of multiple caches, it would have been obvious to store a mapped object in one cache and store the base object in a second cache.

Because we find that the Examiner did not err, we sustain the § 103 rejection of claims 37 and 38 as being unpatentable over Bernstein, Kaakani, and Naga.

CONCLUSION

1. The combination of Bernstein and Kaakani teaches storing a base object handle representing a base object, the mapped object handle comprising a mapping that identifies the base object handle.

2. The combination of Bernstein and Kaakani teaches a mutable mapping type selected from a plurality of mapping types, each of the plurality of mapping types defining semantics of one or more operations on the mapped object.

3. The combination of Bernstein and Kaakani teaches determining one or more actions to perform based on the semantics of the operation defined by the mapping type of the mapped object handle.

4. The combination of Bernstein, Kaakani, and Ozbutun does not teach or suggest a syntactic or semantic mapping type, wherein the base object content includes an evaluation of a function applied to a mapped object handle name.

5. The combination of Bernstein, Kaakani, and Chiu suggests a plurality of mapping types including a versionable type and a non-versionable type.

ORDER

The Examiner's decision to reject claims 1, 12–17, 24–30, 37, and 38 is affirmed.

The Examiner's decision to reject claims 18–23 and 31–34 is reversed.

No time period for taking any subsequent action in connection with this appeal may be extended under 37 C.F.R. § 1.136(a)(1)(iv).

AFFIRMED-IN-PART